IN THE CLAIMS

The claims are as follows:

1. (Previously Presented) In a computer system having a plurality of processors connected across a network to a shared memory, a method of decoupling a write address from its corresponding write data in a store to the shared memory, comprising:

generating a write request address for a memory write, wherein the write request address points to a memory location in the shared memory;

transferring a write request to the shared memory, wherein the write request includes the write request address;

noting the write request address in the shared memory;

comparing, in the shared memory, addresses in subsequent load and store requests to the write request address;

when the corresponding write data becomes available, transferring the write data to the shared memory in instruction order across the network without the write request address;

pairing, within the shared memory, the write request address with the separately transferred corresponding write data; and

storing the write data into the shared memory as a function of the write request address.

2. (Original) The method according to claim 1, wherein the shared memory includes a store address buffer and wherein noting the write request address includes writing the address in the store address buffer.

3. (Previously Presented) The method according to claim 2, wherein comparing addresses in subsequent load and store requests includes stalling subsequent read requests to the write request address until the write data is written into the shared memory.

4. (Original) The method according to claim 1, wherein the shared memory includes a cache, wherein noting the write request address includes changing a state in a cache line

associated with the write request address to "WaitForData", and wherein comparing addresses in subsequent load and store requests to the write request address includes accessing the cache and stalling if a cache line hit returns a "WaitForData" state.

5. (Previously Presented) The method according to claim 1, wherein the shared memory includes a bit vector, wherein noting the write request address in the shared memory includes setting one or more bits in the bit vector corresponding to the write request address, and wherein comparing addresses in subsequent load and store requests to the write request address includes comparing bits that would be set corresponding to the load and store request addresses with the bits set for the write request address and stalling servicing of the load and store requests if there is a match.

6. (Previously Presented) The method according to claim 1, wherein comparing addresses in subsequent load and store requests includes stalling the subsequent load requests to the write request address until the write data is written into the shared memory.

7. (Previously Presented) The method according to claim 6, wherein comparing addresses in subsequent load and store requests includes servicing the load and store requests to addresses other than the write request address without waiting for the write data to be written to the write request address.

8. (Previously Presented) The method according to claim 1, wherein comparing addresses in subsequent load and store requests includes servicing the load and store requests to addresses other than the write request address without waiting for the write data to be written to the write request address.

9. (Previously Presented) The method according to claim 1, wherein comparing addresses in subsequent load and store requests includes enforcing memory ordering in subsequent read and write requests to the write request address until the write data associated with the first write request is written into the shared memory.

10. (Previously Presented) The method according to claim 1, wherein transferring a write request includes ensuring that all vector and scalar loads from shared memory for that processor have been sent to the shared memory prior to issuing the write request.

11. (Previously Presented) In a computer system having a plurality of processors connected to a shared memory, a method of decoupling a write address from its corresponding write data in a write to the shared memory, comprising:

generating a write request address for a memory write, wherein the write request address points to a memory location in shared memory;

issuing a first write request to the shared memory, wherein the first write request includes the write request address;

noting the write request address in the shared memory;

comparing, in the shared memory, addresses in subsequent read and write requests to the write request address;

stalling the subsequent read requests to the write request address until the write data corresponding to the first write request is written into the shared memory; and

if the address in a subsequent write request matches the write request address stored in the shared memory and there are no stalled read requests to the write request address, discarding the first write request.

12. (Original) The method according to claim 11, wherein the shared memory includes a store address buffer and wherein noting the write request address includes writing the address in the store address buffer.

13. (Original) The method according to claim 12, wherein comparing addresses in subsequent read and write requests includes stalling subsequent read requests to the write request address until the write data is written into the shared memory.

14. (Previously Presented) The method according to claim 11, wherein the shared memory includes a cache, wherein noting the write request address includes changing a state in a cache line associated with the write request address to "WaitForData", and wherein comparing addresses in subsequent read and write requests to the write request address includes accessing the cache and stalling if a cache line hit returns a "WaitForData" state.

15. (Previously Presented) The method according to claim 11, wherein the shared memory includes a bit vector, wherein noting the write request address in the shared memory includes setting one or more bits in the bit vector corresponding to the write request address, and wherein comparing addresses in subsequent read and write requests to the write request address includes comparing bits that would be set corresponding to the load and store request addresses the bits set for the write request address and stalling servicing of the load and store requests if there is a match.

16. (Previously Presented) The method according to claim 11, wherein comparing addresses in subsequent read and write requests includes stalling the subsequent read requests to the write request address until the write data is written into the shared memory.

17. (Previously Presented) The method according to claim 16, wherein comparing addresses in subsequent read and write requests includes servicing the read and write requests to addresses other than the write request address without waiting for the write data to be written to the write request address.

18. (Previously Presented) The method according to claim 11, wherein comparing addresses in subsequent read and write requests includes servicing the read and write requests to addresses other than the write request address without waiting for the write data to be written to the write request address.

19. (Previously Presented) The method according to claim 11, wherein comparing addresses in subsequent read and write requests includes enforcing memory ordering in the subsequent read and write requests to the write request address until the write data associated with the first write request is written into the shared memory.

20. (Previously Presented) The method according to claim 11, wherein issuing a write request includes ensuring that all vector and scalar loads from shared memory for that processor have been sent to the shared memory prior to issuing the write request.

21. (Previously Presented) In a computer system having a plurality of processors connected across a network to a shared memory, a method of decoupling a write address from its corresponding write data in a store to the shared memory, comprising:

generating a write request address for a vector store to memory, wherein the write request address points to a memory location in the shared memory;

transferring a vector store request to the shared memory, wherein the write request includes the write request address;

noting the write request address to the shared memory;

comparing, in the shared memory, addresses in subsequent load and store requests to the write request address;

when the corresponding write data becomes available, transferring the write data from a vector register to the shared memory in instruction order across the network without the write request address;

pairing, within the shared memory, the write request address with the separately transferred corresponding write data; and

storing the write data into the shared memory as a function of the write request address.

22. (Original) The method according to claim 21, wherein the shared memory includes a store address buffer and wherein noting the write request address includes writing the address in the store address buffer.

23. (Previously Presented) The method according to claim 22, wherein comparing addresses in subsequent load and store requests includes stalling the subsequent load requests to the write request address until the write data is written into the shared memory.

24. (Original) The method according to claim 21, wherein the shared memory includes a cache, wherein noting the write request address includes changing a state in a cache line associated with the write request address to "WaitForData", and wherein comparing addresses in subsequent load and store requests to the write request address includes accessing the cache and stalling if a cache line hit returns a "WaitForData" state.

25. (Previously Presented) The method according to claim 21, wherein the shared memory includes a bit vector, wherein noting the write request address in the shared memory includes setting one or more bits in the bit vector corresponding to the write request address, and wherein comparing addresses in subsequent load and store requests to the write request address includes comparing bits that would be set corresponding to the load and store request addresses with the bits set for the write request address and stalling servicing of the load and store requests if there is a match.

26. (Previously Presented) The method according to claim 21, wherein comparing addresses in subsequent load and store requests includes stalling the subsequent load requests to the write request address until the write data is written into the shared memory.

27. (Previously Presented) The method according to claim 26, wherein comparing addresses in subsequent load and store requests includes servicing the load and store requests to addresses other than the write request address without waiting for the write data to be written to the write request address.

28. (Previously Presented) The method according to claim 21, wherein comparing addresses in subsequent load and store requests includes servicing the load and store requests to addresses other than the write request address without waiting for the write data to be written to the write request address.

29. (Previously Presented) The method according to claim 21, wherein comparing addresses in subsequent load and store requests includes enforcing memory ordering in the subsequent load and store requests to the write request address until the write data associated with the first write request is written into the shared memory.

30. (Previously Presented) The method according to claim 21, wherein transferring a write request includes ensuring that all vector and scalar loads from shared memory for that processor have been sent to the shared memory prior to issuing the write request.

31. (Previously Presented) A method of decoupling vector data stores from vector instruction execution, comprising:

executing a vector instruction on vector data stored in a vector register, wherein executing a vector instruction includes storing result vector data in the vector register;

generating a vector write address for a vector store;

transferring a vector store request across a network to memory, wherein the vector store request includes the vector write address;

noting the vector write address in the memory;

comparing, in the memory, addresses in subsequent read and write requests to the vector write address;

when the corresponding write data becomes available, transferring result vector data from the vector register to the memory in instruction order across the network without the write request address;

pairing, within the memory, the vector write address with the separately transferred corresponding result vector data; and

storing the result vector data into the memory as a function of the address in the vector store request.

32. (Previously Presented) The method according to claim 31, wherein comparing addresses in subsequent read and write requests to the vector write address includes stalling the subsequent read requests to the vector write address until the result vector data is written into the memory.

33. (Canceled)

34. (Previously Presented) In a processor having a plurality of processing units connected across a network to a shared memory, a method of decoupling a write address from its corresponding write data in a write to the shared memory, comprising:

generating a write request address for a memory write, wherein the write request address points to a memory location in the shared memory;

transferring a write request to the shared memory, wherein the write request includes the write request address;

storing the write request address in the shared memory;

comparing addresses in subsequent read and write requests to the write request address stored in the shared memory;

when the corresponding write data becomes available, transferring the write data to the shared memory in instruction order across the network without the write request address;

pairing, within the shared memory, the write request address with the separately transferred corresponding write data; and

storing the corresponding write data into the shared memory as a function of the write request address.

35. (Previously Presented) The method according to claim 34, wherein transferring a write request includes ensuring that all vector and scalar loads from shared memory for that processor have been sent to the shared memory prior to issuing the write request.

36. (Previously Presented) The method according to claim 34, wherein comparing addresses in subsequent read and write requests includes stalling the subsequent read requests to the write request address until the write data is written into the shared memory.

37. (Previously Presented) The method according to claim 34, wherein comparing addresses in subsequent read and write requests includes enforcing memory ordering in the subsequent read and write requests to the write request address until the write data associated with the first write request is written into the shared memory.

38. (Previously Presented) A computer system, comprising:

a plurality of processors, wherein the processors includes means for issuing a write address separate from data to be written to the write address; and

a shared memory connected by a network to the plurality of processors, wherein the shared memory includes:

means for receiving a first write request including a first write address;

means for noting the write request address to the shared memory;

means for comparing addresses in subsequent load and store requests to the first write address;

means for stalling subsequent load and store requests to a memory location in the shared memory associated with the first write address until the data associated with the first write request is received and written by the shared memory;

means for receiving write data in instruction order across the network without the write request address; and

means for pairing the write request address with the separately transferred corresponding write data prior to storing the write data to the shared memory as a function of the write request address.